|  |  |
| --- | --- |
| Usability Engineering | CS2511 Practical Sessions |

# Dynamic Web Interfaces (ii)

If you completed the last exercise successfully, you should have a web-page that reads an array and, for each element in the array, creates a new object.

The aim of this exercise is to extend the web-page so that objects can be removed as well as added. A control-setting (e.g. a number in a text-box or a value set on a slider) should determine which objects are present at any given time, and changing this value should cause objects to be added to or removed from the page depending upon the new setting.

You may find this exercise easier if you tackle it in stages.

First, modify the array that you created in the last exercise so that it holds both a position and an id for each element.

This will be a multi-dimensional array, which can be created in JavaScript in the following fashion:

|  |  |
| --- | --- |
|  | var object1 = new Array('300px','obj0'); |
|  | var object2 = new Array('400px','obj1'); |
|  | (etc.) |
|  | var objects = new Array(object1,object2,...); |

Since the array is now multi-dimensional, it's necessary to distinguish between the different pieces of information stored for each object. This is done using a second index:

* The first index selects a sub-array (object1 or object2 in the example above)
* The second index selects an element in that sub-array (the position or the id in the example above)

For example:

objects[0][1]

would select the first sub-array (object1) and the second element in that sub-array (the string obj0).

Modify the remainder of the code from the last exercise so that, when elements are created, both the position and the id are obtained from the array.

When this is working correctly, create a button that deletes one of the objects. To remove an object you can use removeChild(), e.g.:

documentBody.removeChild(document.getElementById('obj1'));

Where:

* documentBody is a reference to the document element, obtained in the same way as in the last exercise.
* obj1 is the id of an element on the page.

Check that calling this function removes the specified object.

When this is working correctly, extend the function so that it deletes ALL the objects. It should loop through the array, obtaining each id in turn and then removing the object with that id.

The next stage is to modify the code so that objects are only created if one of their properties falls above or below a specified threshold. As an example, add a 'price' value to each element, and then set a threshold so that only items priced less than this threshold are displayed.

First, extend the array so that it holds three items of data for each element - the position and id, plus a numerical value which will represent the price. Give each element a different price.

Add a text-box or similar input widget so that you can set the threshold value (the maximum price).

Now add a conditional statement to the 'create' code so that elements are only created when the third value in the array (the price) is below the threshold set in text-box.

Similarly, modify the 'delete' code so that elements are only deleted when they are above the threshold value.